How effective are machine learning regression models at analyzing stock exchange data?

Matthew Auger

December 10, 2022

1 Introduction

1.1 Motivation & Goals

The stock market offers opportunities for businesses and investors to grow their wealth. By buying and selling stocks, investors can take advantage of market movements to their benefit. However, the stock market can be risky. To make informed decisions and reduce this risk, investors use tools and techniques for analyzing stock market data and predicting future movements.

Machine learning and AI are powerful tools for stock market analysis, offering the potential to automate and improve analysis processes. In this report, I will explore the effectiveness of machine learning models in analyzing stock exchange data, focusing on regression models by analyzing the NYA index. By evaluating these techniques, I aim to offer insights into their potential for stock market analysis, and determine if regression is a suitable analysis technique.

1.2 Review of existing literature

Regression techniques in general, work by taking a data set of one or more independent variables and try to best predict what a dependant variable would be by plotting a line, curve or plane to fit the data. Linear and multiple linear models work by taking the independent variables and by using training data, try to fit a function of the form $y = B_0 + B_1x_1 + B_2x_2 + ... + E$, where B_n are the independent coefficients that represent the weight of each variable x_n and E is the measure of random error in the model. Linear regression only goes up to $n = 1$. [1]

Polynomial regression models operate on a similar basis, but instead tries to find a function of the form $y = B_0 + B_1x + B_2x^2 + B_3x^3 + ... + B_nx_n^n$ up to degree n which is a parameter that needs tuning when developing the model. Polynomial techniques are more sensitive to normalisation, and require more tuning to get the parameters right to prevent over fitting. As well they can require a large training set to get accurate predictions. However after this their predictions are generally more accurate provided they don't diverge from the data [2]

2 Data & Methodologies

2.1 Description of the data set

The data set I'm working with came from kaggle.com [3] and contains daily price data for stock market indexes. The data is presented in a CSV file, which includes daily prices from 1965 to 2021 for multiple stock exchanges, such as NYA, IXIC, and GSPTSE. The data set includes attributes Index, Date, Open, High, Low, Close, Adj Close, and Volume. This provides a range of daily information that can be analyzed and used in the regression models I wish to train, though not all of the attributes are useful. For example, the "Adj Close" attribute is similar to the "Close" attribute and doesn't provide enough useful information so it gets removed during data cleaning. In addition, some entries in the data set contain NaN values, indicating missing or unavailable data. These entries will need to be excluded too.

2.2 Methodology

Before the data can be used in the models, it needs to be processed and cleaned. First of all I will read the CSV file into a pandas DataFrame so it can be easily manipulated. As I'm only looking at the NYA exchange the other stock exchanges data can be discarded. In addition the amount of data available in this data set, almost 6 decades worth of data, is excessive for this report. Thus I selected data within in a 6 year span from the start of 2010 to the end

of 2015. This was an arbitrary choice by me and any time span could've been selected. In order to input this time data into the models, it needs to be converted from strings into either datetime objects or as integer number of days since a chosen day. The NaN values that appear throughout the data need eliminating too. [1] suggests using fillna() to estimate the data however I'm choosing to remove the rows with the dropna() function because it is a simpler option. Although not an issue in this case, this could cause issues as a solution if data contains a large enough number of NaN values. Finally to get my final dataset, I will drop features which I will not be using to train my models. Specifically: Index, Adj Close and Volume attributes will be dropped leaving only; Date, Open, High, Low and Close.

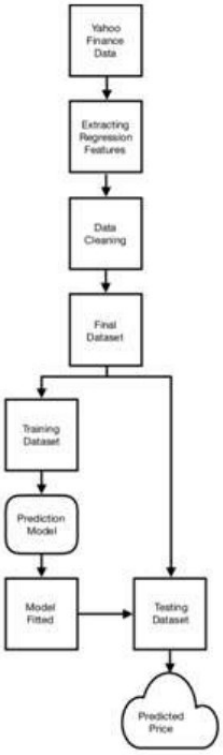

Figure 1: Multiple Linear Regression Methodology from [1]

Once the data has finally been processed I have my final data set. For linear and Multiple linear regression, I will be following a similar methodology of [1]. Figure 1 comes from this paper, showing the steps taken as a flowchart. For both I will be training the models to predict the stock's Close price, using the Open, High and Low prices as data. When splitting the data into training and testing sets, I will split the data as 70% training and 30% testing. For polynomial regression I will be trying to recreate something similar to [2] with support vector regression to train a model on the historical data and to create a polynomial curve to predict the trend of the markets in the future. Both methods can be implemented with python by using the sklearn library for the regression models and then other tools such as matplotlib and numpy to graph and process the data into a more readable state. My methods will be included in a jupyter notebook with this report.

3 Results

3.1 Summary of the modelling outcome

3.1.1 Linear Regression

	Model Coefficients
Open	0.999665
High	1.004513
Low	0.992934

Figure 2: Individual Model Coefficients for each model

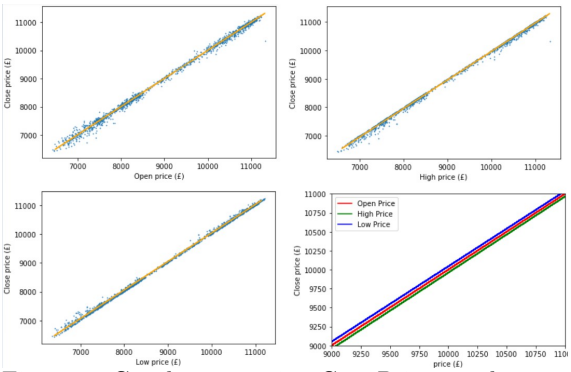

Figure 3: Graphs comparing Cost Price to whatever the model is. Graph 4 plots the regression lines of the other graphs

For this technique, I trained 3 models to predict the Close Price. One for each independent variable in my data set excluding the date. They all produced similar looking results, each with a Model Coefficient valued around 1, as seen on figure 2. This means they had functions approximate to $y = x$, which can be seen in figure 3, plotting the variable to the Close Price. This indicates the models were effective in predicting the Close price based off their variable of that days stock. When we plot the predicted Close Price with time it looks similar to the true Close Price graph, with few noticeable deviations, per figure 4.

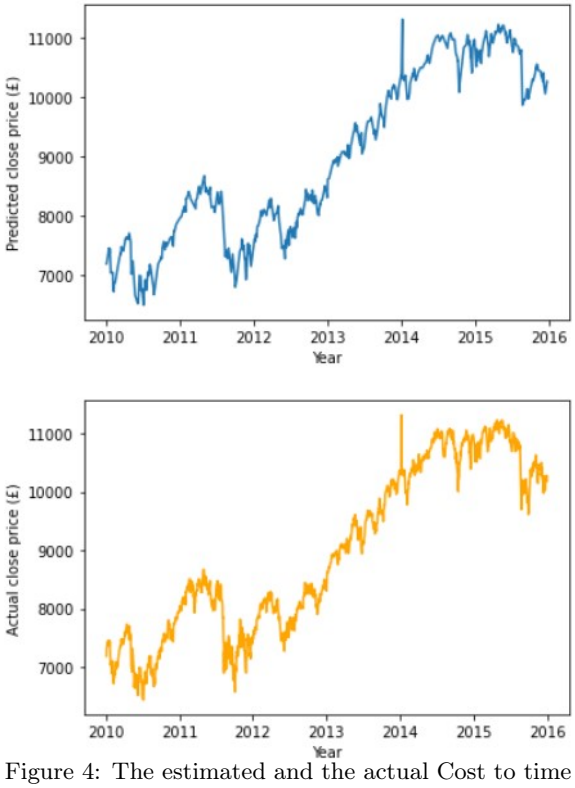

Figure 4: The estimated and the actual Cost to time

3.1.2 Multiple Linear Regression

	ML Model Coeeficients
Open	-0.594301
High	0.800939
Low	0.791665

Figure 5: Coefficients for the Multiple Linear Model

In contrast, I only trained one model for Multiple Linear Regression as it combines the statistical importance of the three independent variables into one function. As expected, the Model Coefficients were much more varied, see figure 5.

This shows that the High and Low price variables had a greater statistical effect in the prediction of Close Price than the Open variable.

3.1.3 Polynomial Regression

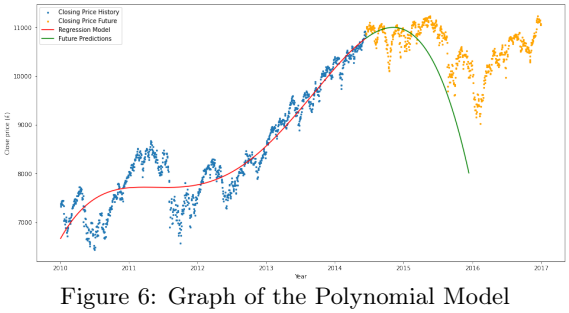

Figure 6: Graph of the Polynomial Model

The process for Polynomial Regression was more complex, even though it was one model to be trained. Future Dates beyond what was used to train the model had to be saved so the model's prediction could be tested. The model had to be trained using polynomial features instead of linear, and by trial and error I determined a fourth order polynomial was the best way to model the trend of the training set. Per figure 6, The model of the Historical date fits very well, staying within a reasonable average the whole time. Initially too the prediction works well, showing a downward trend is coming which is seen in the future data. However the price recovers and increase again in the future and the model keeps indicating a decrease in price, showing the limit in it's predictive ability.

3.2 Comparison and evaluation of the models

	Variable	RMSE	MAE	R2
0	Open	88.575795	57.319123	0.995854
1	High	73.263255	40.947893	0.997301
2	Low	51.651556	35.620734	0.998585
3	MLR	2109.301260	1701.349629	-1.162010

Figure 7: table listing errors

The linear and multiple linear models produced largely the same results, especially when comparing the final graphs produced. A large difference though is that the multiple linear model had a lot more error in its date, as shown in figure 7. I believe the reason for this is because the multiple linear model has to consider the 3 inputs at once, the uncertainty in each individual variable skewing the total, where the individual linear models only had 1 variable's uncertainty which wasn't being affected by other variables. Figure 7 also shows that Open Price is the most uncertain variable for predicting Close Price and Low was the best.

Neither of this models however are capable of predicting more than the Close Price on a given day, unlike the polynomial model. As seen on figure 6, the model is easily capable of following trends in stock indices and is capable of making sound predictions on what will happen at least 6 months into the future. Any further than this however and it starts to become too inaccurate.

Overall it seems that for daily predictions, simple linear models have the least error in results, especially if you use High or Low Price variables, so the extra accuracy would be beneficial compared to multiple linear or polynomial models. But for mid to long term analysis of trends, polynomial models become more suitable

as they are easier to interpret and the predictions are still reasonably accurate.

4 Discussion

4.1 Summary of key findings

Overall, we found that regression models can be used in a range of ways to try analyse stock data. Linear and Multiple Linear Regression models can be used to analyse what the closing value of a stock index will be in a day, the most effective of these being the linear model using the Low Price to generate predictions. Also with the amount of data we used, Linear models were much more accurate than Multiple Linear models, making them much more effective. Finally if we wish to analyse further into the future then polynomial models are effective at predicting the general trend up to around 6 months after which they become ineffective.

4.2 H - What is next in modelling the data, how can the methods be improved

To further improve the models, I theorize that for Multiple Linear Regression, feeding more data will help tune the models leading to more accurate results that can be used in more applications. As for polynomial regression, the obvious improvement is making the predictions go further into the future by tuning how much training data it receives and what order polynomial you model as. Although this is limited to a point as if the order is too high, the model over fits.

References

[1] S. Shakhla, B. Shah, N. Shah, V. Unadkat, and P. Kanani, "Stock price trend prediction using multiple linear regression," *Int. J. Eng. Sci. Inven.(IJESI)*, vol. 7, no. 10, pp. 29–33, 2018.

[2] L. Nunno, "Stock market price prediction using linear and polynomial regression models." [Online]. Available: http://www.lucasnunno.com/assets/docs/ml_paper.pdf

[3] kaggle. [Online]. Available: https://www.kaggle.com/datasets/mattiuzc/stock-exchange-data